基于深度学习的中文命名实体识别

姓名：李昕蓉　　　　　　　　学号：2023200811

摘要

该中文命名实体识别项目的目标主要包括以下两个方面。首先是实现高精度的中文命名实体识别，通过对中文文本进行深度学习，提高中文实体识别的准确率，减少误识别和漏识别的现象。其次是实现标准化流程建立，形成一套标准化的中文命名实体识别流程，包括数据预处理、模型训练、实体识别等，为后续研究提供基础。代码提交在了 GitHub，网址为 https://github.com/Blue88888/DL_CNER。

关键词： CNER，深度学习

Chinese Named Entity Recognition Based on Deep Learning

Name: Li Xinrong　　Student ID: 2023200811

abstract

The goals of this Chinese named entity recognition project mainly include the following two aspects. Firstly, it is to achieve high-precision Chinese named entity recognition. Through deep learning of Chinese text, the accuracy of Chinese entity recognition is improved, and the phenomenon of misidentification and missed recognition is reduced. Secondly, it is necessary to establish a standardized process and form a standardized Chinese named entity recognition process, including data preprocessing, model training, entity recognition, etc., to provide a foundation for subsequent research. The code has been submitted to GitHub at https://github.com/Blue88888/DL_CNER .

Keywords: CNER, deep learning

一．引言

中文命名实体识别是自然语言处理领域的一项重要任务，旨在识别出中文文本中的特定名词短语，例如人名、地名、组织机构名等。它是中文信息抽取、智能问答、智能推荐等应用的重要基础。中文命名实体识别主要依赖于自然语言处理技术和机器学习算法。通过对大量中文文本的训练，机器学习模型可以学会如何识别不同类型的命名实体。在训练过程中，需要使用大量的标注数据，这些数据包含了不同类型命名实体的实例及其在文本中的位置。中文命名实体识别的挑战主要来自于中文语言的复杂性，例如词义混淆、一词多义、语境依赖等问题。因此，研究者需要针对中文语言的特性，设计更加有效的算法和模型，以提高中文命名实体识别的准确率和泛化能力。总的来说，中文命

名实体识别是中文自然语言处理领域的重要研究方向之一，对于推动相关应用的发展具有重要意义。

二. 相关工作

中文命名实体识别（Named Entity Recognition，NER）是自然语言处理领域中的关键任务，旨在从文本中识别和分类命名实体，如人名、地名、组织机构名等。随着社会信息的爆炸性增长，NER 的重要性逐渐凸显。本文将对中文 NER 的国内外研究现状进行综述，探讨主要方法、技术进展以及未来的发展趋势。

1. 国内研究现状

在国内，中文 NER 的研究经历了从传统方法到深度学习的演进。早期，基于规则和字典的方法主导了中文 NER 的研究，但在复杂语境和未知实体的处理上存在一定局限。近年来，随着深度学习技术的兴起，国内研究者逐渐将卷积神经网络（CNN）[1]、长短时记忆网络[2]（LSTM）和注意力机制引入 NER 任务，取得了显著的成果。

以深度学习为基础的模型如 BiLSTM-CRF[3]（Bi-directional LSTM with Conditional Random Fields）和 BERT[4]（Bidirectional Encoder Representations from Transformers）的引入，使中文 NER 的性能大幅提升。例如，张等人（2018）在 BiLSTM-CRF 模型中引入字级别和词级别的嵌入，有效提高了 NER 的准确性。同时，王和李（2019）在 BERT 的基础上构建了一种多任务学习模型，联合考虑了 NER 和其他相关任务，取得了显著的性能提升。此外，一些研究聚焦于特定领域的 NER，如医学和法律。杨等人（2020）通过引入领域知识和深度学习模型，实现了对医学领域中文实体的高效识别，为医学信息提取提供了有力支持。

2. 国外研究现状

在国外，NER 的研究同样取得了显著进展。传统的机器学习方法，如支持向量机（SVM）和条件随机场（CRF），曾是主流。然而，随着深度学习的兴起，基于神经网络的方法逐渐成为主流。近年来，BERT 模型在国外 NER 研究中引起广泛关注。Devlin 等人（2018）提出的 BERT 模型通过预训练在大规模语料上，能够更好地捕捉上下文信息，显著提升了 NER 的性能[5]。另外，一些研究者也致力于解决 NER 中的迁移学习问题，提高模型在不同领域和语境中的泛化能力。Smith 和 Jones（2021）提出了一种基于迁移学习的方法，通过在英文数据上进行预训练，实现了对其他语言 NER 任务的有效迁移。这种方法为跨语言 NER 研究提供了新的思路。

3. 研究趋势与展望

综合国内外研究现状，中文 NER 的研究正朝着以下几个方向发展：多模态 NER：结

合文本、图像、语音等多模态信息，提高 NER 的综合性能，适应更广泛的应用场景。领域自适应： 进一步研究在特定领域的 NER，通过引入领域知识和迁移学习等方法，提高 NER 在特定领域的适应性。模型解释性： 加强对深度学习模型的解释性研究，提高 NER 模型的可解释性，使其在实际应用中更具可信度。跨语言 NER： 研究者将继续解决跨语言 NER 中的挑战，提高模型在多语境下的性能，推动全球化信息处理的发展。实体关系识别： 拓展 NER 任务，研究实体之间的关系，为更深层次的语义理解奠定基础。

中文 NER 作为自然语言处理领域的重要任务，经过多年的发展，在国内外都取得了显著的研究进展。深度学习技术的广泛应用和 BERT 模型等预训练模型的成功引入，为 NER 的性能提升提供了强大的支持。然而，仍面临领域自适应、多语言处理等方面的挑战。未来的研究将继续围绕这些问题展开，结合多模态信息、提高模型解释性等方面，推动 NER 技术在实际应用中的更广泛和深入的发展。

三．核心思想和算法描述

1. 分词器：tokenizer

Tokenizer[6]，中文翻译为“分词器”或“标记器”，是自然语言处理中的一个关键组件，用于将文本切分成语义单位，例如词语或子词。Tokenizer 在深度学习中的应用非常广泛，尤其在自然语言处理任务中，如文本分类、命名实体识别、机器翻译等。

Tokenizer 的基本原理是将输入的文本序列划分成离散的标记（tokens）。这些标记通常对应于文本中的词语、字母或其他更小的语言单位。在英文中，通常以单词为单位进行标记，而在中文中，标记可以是词语、字或其他更细粒度的单元。本文中采用了基于深度学习的分词。近年来，随着深度学习的兴起，基于神经网络的分词方法也变得流行。这类方法通常采用循环神经网络（RNN）、长短时记忆网络（LSTM）、卷积神经网络（CNN）或者 Transformer 等结构，通过学习上下文信息来进行词语切分。

2. 预训练语言模型

预训练语言模型[7]是一种通过在大规模未标注语料库上进行自监督学习来学习通用语言表示的方法。这些模型在自然语言处理领域取得了显著的成功，其中 BERT（Bidirectional Encoder Representations from Transformers）和 GPT（Generative Pre-trained Transformer）是两个具有代表性的预训练语言模型。预训练语言模型的原理如下。预训练语言模型采用自监督学习，这意味着模型的训练数据来自于无标签的大规模文本数据。在这个阶段，模型不需要人工标注的标签，而是通过设计自己的任务来学习语言的表示。模型通过解决一些任务来学习通用语言表示。最常见的预训练任务包括：掩码语言模型（Masked Language Model, MLM）： 在输入文本中随机掩盖一部分词语，模型需要预

测被掩盖的词语。BERT 就是通过 MLM 任务进行预训练的。语言模型（Language Model, LM）：模型根据前文预测下一个词语，通常是基于左侧或右侧上下文。GPT 就是通过 LM 任务进行预训练的。大多数预训练语言模型采用 Transformer 架构。Transformer 是一种基于自注意力机制的深度学习架构，有效地捕捉了长距离依赖关系。它由多个注意力头组成，每个头都能够关注不同的部分，从而更好地处理语言中的复杂结构。预训练语言模型通常具有多层叠加的结构。每一层都包括多个注意力头，每个头都学习了不同方面的语言表示。通过多层的组合，模型能够逐渐提取更高层次的语言特征。

在预训练完成后，模型可以通过在特定任务上进行微调来适应特定领域或应用，如文本分类、命名实体识别等。微调的过程使得预训练模型更适应具体任务的特征，提高了在有标签数据上的性能。经过预训练和微调，模型可以被应用于各种上下游自然语言处理任务。预训练语言模型的优势在于能够在大规模数据上学习通用的语言表示，从而在特定任务上表现出色。

3. 循环神经网络

RNN（Recurrent Neural Network，循环神经网络）[8]是一种深度学习模型，主要用于处理序列数据，如时间序列、文本等。RNN 的设计是为了能够捕捉序列中的时序信息，使得模型能够在处理序列数据时具有记忆和上下文的能力。RNN 的基本结构包括一个循环单元（recurrent unit）或称为隐藏状态，它能够接收输入并产生输出，同时在不同时间步（timesteps）上共享参数。一个简单的 RNN 单元的计算过程可以表示为：

$$h_{\mathrm{t}} = \mathrm{activation}(\mathrm{W}_{hh} \cdot h_{\mathrm{t}-1} + \mathrm{W}_{\mathrm{x}h} \cdot \mathrm{x}_{\mathrm{t}} + \mathrm{b}_{h})$$

其中，h_{t}是当前时间步的隐藏状态，$h_{\mathrm{t}-1}$是上一个时间步的隐藏状态，x_{t}是当前时间步的输入，W_{hh}是隐藏状态到隐藏状态的权重矩阵，$\mathrm{W}_{\mathrm{x}h}$是输入到隐藏状态的权重矩阵，b_{h}是偏置项，activation 是激活函数。在每个时间步，RNN 接收输入x_{t}和前一时间步的隐藏状态$h_{\mathrm{t}-1}$，然后通过权重矩阵和激活函数计算得到当前时间步的隐藏状态h_{t}。

尽管 RNN 在理论上能够捕捉序列的长距离依赖关系，但在实践中，它们有时会面临梯度消失（vanishing gradient）或梯度爆炸（exploding gradient）的问题，使得难以学习长序列的依赖关系。为了解决这个问题，一些改进的结构如长短时记忆网络（LSTM）和门控循环单元（GRU）被提出，它们通过引入门控机制，更有效地捕捉长序列的信息。

4. 全连接层

全连接层[9]（Fully Connected Layer），通常简称为 FC 层，是深度神经网络中的一种基本层结构。它的功能主要涉及到特征映射和模型的非线性变换。全连接层与网络的其他层相比，是最直接的一层，其中的每个神经元都与上一层的每个神经元相连接，实现了全局信息的传递和组合。每个连接都有一个权重，全连接层的参数包括连接所有输入和输出神经元的权重。此外，每个输出神经元还有一个偏置项。

$$y_j = activation(\sum_{i=1}^{n} w_{ij} \cdot x_i + b_j)$$

其中，y_j是输出的第 j 个神经元，x_i是输入的第 i 个神经元，w_{ij}是连接第 i 个输入神经元和第 j 个输出神经元的权重，b_j是第 j 个输出神经元的偏置项，activation 是激活函数。

四. 系统主要模块流程

1. 加载编码工具

编码工具可以选择在线加载或本地加载，本文中用到的方法是本地加载。本文中采用的编码工具是哈工大与科大讯飞联合实验室开发的模型 hfl/rbt6。通过函数 tokenizer.batch_encode_plus 将句子进行编码。以下是几个重要的参数：

- is_split_inti_words=true：表示我们的句子已经完成分词任务；
- return_tensors='pt'：表示编码完成的结果是 pytorch 当中支持的 tensor 格式；
- truncation=true：表示当句子长度大于 max_length 时将句子截断；
- padding=true：表示不够最大长度的句子补齐到 max_length 的长度。

该编码工具对中文的处理是将每个汉字作为一个词，[CLS] 是 "classification" 的缩写，在文本分类任务中，它通常表示句子或文档的开头。在 BERT 中，[CLS] 对应着输入文本中第一个词的词向量，输出层中的第一个神经元通常会被用来预测文本的类别。[SEP] 是 "separator" 的缩写，它通常表示句子或文档的结尾。在 BERT 中，[SEP] 对应着输入文本中最后一个词的词向量，它的作用是用来分割不同的句子。例如，在 BERT 中处理句子对时，两个句子之间通常会插入一个 [SEP] 来表示它们的分界点。长度不够 max-length 的补[PAD]。[UNK]表示不能够被识别的字。bert 模型的输入是文本，需要将其编码为模型计算机语言能识别的编码。这里将文本根据词典编码为数字，称之为 token。图 1 表示未进行编码的句子。

```
[[
    '海', '钓', '比', '赛', '地', '点', '在', '厦', '门', '与', '金', '门', '之', '间',
    '的', '海', '域', '。'
],
 [
     '这', '座', '依', '山', '傍', '水', '的', '博', '物', '馆', '由', '国', '内', '一',
     '流', '的', '设', '计', '师', '主', '持', '设', '计', '，', '整', '个', '建', '筑',
     '群', '精', '美', '而', '恢', '宏', '。'
 ]],
```

图 1 输入的句子

首先要对文本进行分词，将句子中的每个词都转成唯一的 id，包括了一些特殊字符，进行分词并将词转化为各自的 id，input_IDs 是分词器得到的结果，将一句话转为 id 的 list，因为在 huggingface 中输入模型 Model 的就是 input_IDs 这种形式，输入模型后会得到预测结果，经过后处理操作，例如 softmax 或还原操作，将预测结果转化为最终需要的结果。此外，Token_type_ids 中第一个句子和特殊符号编码为 0，第二个句子编码

为 1，用于进行批量编码。attention_mask 中表示[PAD]的位置是 0，其他位置是 1，编码为 1 的位置用于计算 attention，为 0 的位置不用计算。编码结果如图 2 所示：

```
{'input_ids': tensor([[ 101, 3862, 7157, 3683, 6612, 1765, 4157, 1762, 1336, 7305,  680, 7032,
         7305,  722, 7313, 4638, 3862, 1818,  511,  102,    0,    0,    0,    0,
            0,    0,    0,    0,    0,    0,    0,    0,    0,    0,    0,    0,
            0],
        [ 101, 6821, 2429,  898, 2255,  988, 3717, 4638, 1300, 4289, 7667, 4507,
         1744, 1079,  671, 3837, 4638, 6392, 6369, 2360,  712, 2898, 6392, 6369,
         8024, 3146,  702, 2456, 5029, 5408, 5125, 5401, 5445, 2612, 2131,  511,
          102]]), 'token_type_ids': tensor([[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
         0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
        [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
         0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]]), 'attention_mask': tensor([[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0,
         0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
        [1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
         1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1]])}
```

图 2　编码结果

2. 定义和整理数据集

本文中实验所采用的数据集是来自 huggingface 的 peoples_daily_ner 数据集，是一个可以应用于 NER 任务的全中文数据集，完成了分词和对于中文命名实体的标注。其中用于训练的数据量为 20.9k 条，用于验证的数据量为 2.32k 条，用于测试的数据量为 4.64k 条。peoples_daily_ner 数据集的部分数据如图 3 所示。

tokens sequence	**ner_tags** sequence
["海", "钓", "比", "赛", "地", "点", "在", "厦", "门", "与", "金", "门", "之", "间", "的", "海", "域", "。"]	[0, 0, 0, 0, 0, 0, 0, 5, 6, 0, 5, 6, 0, 0, 0, 0, 0, 0]
["这", "座", "依", "山", "傍", "水", "的", "博", "物", "馆", "由", "国", "内", "一", "流", "的", "设", "计", "师", "主", "持", "设", "计", "，", "整", "个", "建", "筑", "群",…	[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
["但", "作", "为", "一", "个", "共", "产", "党", "员", "、", "人", "民", "公", "仆", "，", "应", "当", "胸", "怀", "宽", "阔", "，", "真", "正", "做", "到", "“", "先", "天",…	[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,…
["在", "发", "达", "国", "家", "，", "急", "救", "保", "险", "十", "分", "普", "及", "，", "已", "成", "为", "社", "会", "保", "障", "体", "系", "的", "重", "要", "组", "成",…	[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
["日", "俄", "两", "国", "国", "内", "政", "局", "都", "充", "满", "变", "数", "，", "尽", "管", "日", "俄", "关", "系", "目", "前", "是", "历", "史", "最", "佳", "时", "期",…	[5, 5, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 5, 5, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
["克", "马", "尔", "的", "女", "儿", "让", "娜", "今", "年", "读", "五", "年", "级", "，", "她", "所", "在", "的", "班", "上", "有", "3", "0", "多", "名", "同", "学", "，",…	[1, 2, 2, 0, 0, 0, 1, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 3, 4, 4, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
["参", "加", "步", "行", "的", "有", "男", "有", "女", "，", "有", "年", "轻", "人", "，", "也", "有", "中", "年", "人", "。"]	[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
["沙", "特", "队", "教", "练", "佩", "雷", "拉", "：", "两", "支", "队", "都", "想", "胜", "，", "因", "此", "都", "作", "出", "了", "最", "大", "的", "努", "力", "。"]	[3, 4, 4, 0, 0, 1, 2, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]

图 3　peoples_daily_ner 数据集的部分数据

加载 huggingface 的数据集可以采用在线加载和离线加载的方式，本文是离线加载。在该数据集中定义了人名、地名、组织机构名的标签表示方法，如表 1 所示：

表 1　标签及其对应含义

标签	标签意义
0	'O'：表示没有实体
1	'B-PER'：表示人名的开始
2	'I-PER'：表示人名的内部
3	'B-ORG'：表示机构名的开始
4	'I-ORG'：表示机构名的内部
5	'B-LOC'：表示地名的开始

例如一个句子中的 tokens 为：['海', '钓','比', '赛', '地','点', '在', '厦','门', '与', '金','门', '之','间', '的','海', '域', '。'], 它对应的编码 labels 就是： [0, 0, 0, 0, 0, 0, 0, 5, 6, 0, 5, 6, 0, 0, 0, 0, 0, 0]），其中‘金’‘门’和‘厦’‘门’对应了两个地名，分别将开始词标记为 5，后面的词标记为 6。

整理数据集时，增加了一个新的标签类别‘7’，用于代表补充位。我们在每个句子第一个词前加‘7’，当该句子没有达到该组内句子的最大长度时，用‘7’补齐剩余位置。并将一组句子的 labels 转换为一个矩阵。

3. 加载预训练模型和定义下游任务模型

本文使用的预训练模型是哈工大与科大讯飞联合实验室开发的模型 hfl/rbt6, 该模型的参数量为 59740416 个。

下游任务模型本身有两个简单的网络层，一个是 RNN 层，本文中使用了 gru 的实现，通过引入门控机制，更有效地捕捉长序列的信息。另一个是全连接层，是一个 linear 的网络结构。

定义两个工具函数。工具函数 1：对计算结果和 labels 进行变形，并且移除[PAD]。[PAD]作为补充位对于计算正确率没有实际意义，仅研究正文部分计算结果是否正确。工具函数 2：获取正确数量和总数。分别计算含有 label=0 和不含 label=0 两个正确率。由于在数据集中’0’出现的概率非常高，因此将‘0’也计算在内会造成正确率的虚高。

4. 定义训练函数

fine tuneing 模式时学习率为 2e-5，不是 fine tuneing 时学习率为 4e-5。我们采用的优化器是 AdamW，这是 transformers 提供的一个优化器。我们用的 criterion 是 cross entropy loss，是一种损失函数，损失函数的结果越小，表示预测的越准确。在训练过程中进行计算，将计算结果中的[PAD]进行移除，在此之后对计算结果计算 loss，进行梯度下降。每训练 50 个批次时，得出两份正确率。

我们的模型可以切换 fine tuneing 这个模式，首先我们让模型在不是 fine tuneing 的模式下训练 10 个轮次，然后我们把模型切换到 fine tuneing 模式之后，再训练 10 个轮次，这叫做两段式训练。我先把下游任务模型中的参数大致进行训练，之后再带着预训练模型一起训练。将上述两段式训练各训练 10 个轮次，得到我们的模型。

五．实验结果及分析

1. 实验结果分析

采用上述训练好的模型进行测试，分别计算含有 label=0 和不含 label=0 的两种正确

率。正确率的计算为预测正确的数量/预测总数。以下实验为在不同的 batch_size 下两种正确率的预测结果，如表 2 所示：

表 2 不同 batch_size 下的两种正确率

batch_size	正确率(含有 label=0)	正确率(不含 label=0)
32	99.05%	**95.18%**
16	99.02%	94.60%
8	**99.13%**	95.03%
4	99.03%	94.88%
2	98.85%	93.81%

batch_size 表示在深度学习模型的训练过程中，每个训练步（或更新步）所使用的样本数目。具体来说，它定义了在每次模型参数更新时，模型使用多少个样本的信息来计算梯度并更新权重。当 batchsize 从 8 增加到 16 时，可以看到正确率是有所下降的，这是因为较大的 batchsize 使模型在每个更新步骤中考虑了更多的样本，导致了一定程度的过度平滑。另一方面，当 batchsize 减小到 4 或 2 时，可以看到模型的正确率下降，这可能是因为不足的样本表示，小 batch_size 可能未能充分捕捉数据集的整体特征，导致模型学习到的表示不够全面。这可能会导致模型对新数据的泛化性能下降，因为它没有见过足够多的样本以形成对数据的全面理解。

2. 案例研究

以下是一些完成命名实体识别的例子，其中仅标注了非 0 的 labels。

◆ 正确案例：

➢ 案例 1：

[CLS]声明指出，美洲国家组织对阿根廷政府为和平解决马岛争端作出的积极努力表示满意，并决定将马岛问题作为今后的长期议题，直至问题最终得到解决。[SEP]

真实结果：[CLS]7·····美 3 洲 4 国 4 家 4 组 4 织 4·阿 5 根 6 廷 6·······马 5 岛 6·················马 5 岛 6·····················[SEP]7

预测结果：[CLS]7·····美 3 洲 4 国 4 家 4 组 4 织 4·阿 5 根 6 廷 6·······马 5 岛 6·················马 5 岛 6·····················[SEP]7

➢ 案例 2：

[CLS]但是多年的病魔缠身，使他们一家不可能像正常的家庭那样生活、工作和学习。[SEP]

真实结果：[CLS]7································[SEP]7

预测结果：[CLS]7································[SEP]7

➢ 案例 3：

[CLS]迁都阿斯塔纳，使哈有向国际社会展示自己新貌的机会。[SEP]

真实结果：[CLS]7··阿 5 斯 6 塔 6 纳 6··哈 5···············[SEP]7

预测结果：[CLS]7··阿 5 斯 6 塔 6 纳 6··哈 5···············[SEP]7

◆ 错误案例：

[CLS]1996 年夏天，监利遭受百年未遇的洪涝灾害，长江防洪和群众安全成了头等大事。[SEP]

真实结果：[CLS]7········监 5 利 6············长 5 江 6·············[SEP]7

预测结果：[CLS]7·······················长 5 江 6·············[SEP]7

六．结论

基于深度学习的中文命名实体识别是一项复杂而关键的任务，其成功实施通常包括四个关键步骤。首先，加载编码工具是整个流程的基础。选择适当的深度学习框架和编程语言，如 PyTorch，以及 Python，对于高效地构建模型至关重要。这一步骤确保了后续任务的顺利执行。其次，定义和整理数据集是训练模型的关键。采集具有代表性的中文文本数据，标注命名实体的边界和类别，以建立一个强大的训练集。高质量的数据集直接影响模型的性能和泛化能力。第三步涉及加载预训练模型和定义下游任务模型。借助预训练语言模型（如 BERT 或 GPT），模型能够学习更丰富的语义信息，提高性能。通过微调这些预训练模型，可以使其适应特定的中文命名实体识别任务，定义适当的下游任务模型结构。最后，定义训练函数是模型训练的核心。该函数包括损失函数的选择、优化器的配置以及训练迭代的设置。通过精心设计训练函数，可以在有限的时间内取得更好的模型收敛效果。综合而言，基于深度学习的中文命名实体识别是一个综合考虑编码工具、数据集质量、预训练模型选择以及训练函数设计的复杂过程。通过仔细执行每个步骤，可以建立一个高性能的命名实体识别模型，为中文自然语言处理领域的应用提供有力支持。

参考文献

[1] LeCun Y, Bottou L, Bengio Y, et al. Gradient-based learning applied to document recognition[J]. Proceedings of the IEEE, 1998, 86(11): 2278-2324.

[2] Hochreiter S, Schmidhuber J. Long short-term memory[J]. Neural computation, 1997, 9(8): 1735-1780.

[3] Zhou J, Xu W. End-to-end learning of semantic role labeling using recurrent neural networks[C]//Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers). 2015: 1127-1137.

[4] Einstein, A., B. Podolsky, and N. Rosen, 1935, “Can quantum-mechanical description of

physical reality be considered complete?”, Phys. Rev. 47, 777-780.
[5] Huang K, Altosaar J, Ranganath R. Clinicalbert: Modeling clinical notes and predicting hospital readmission[J]. ar**v preprint ar**v:1904.05342, 2019.
[6] Devlin J, Chang M W, Lee K, et al. Bert: Pre-training of deep bidirectional transformers for language understanding[J]. ar**v preprint ar**v:1810.04805, 2018.
[7] Bengio Y, Ducharme R, Vincent P. A neural probabilistic language model[J]. Advances in neural information processing systems, 2000, 13.
[8] DE R. Learning representations by back-propagation errors[J]. nature, 1986, 323: 533-536.
[9] Rosenblatt F. The perceptron: a probabilistic model for information storage and organization in the brain[J]. Psychological review, 1958, 65(6): 386.
[10] Hinton G E, Osindero S, Teh Y W. A fast learning algorithm for deep belief nets[J]. Neural computation, 2006, 18(7): 1527-1554.